BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

Requested Patent: JP11085519A

Title:

TECHNIQUE FOR PROGRAMMATICALLY CREATING DISTRIBUTED OBJECT PROGRAMS ;

Abstracted Patent: US6157960 ;

Publication Date: 2000-12-05 ;

Inventor(s):

KING RICHARD ADAM (US); LI ZHIYONG (US); KAMINSKY DAVID LOUIS (US) ;

Applicant(s): IBM (US) ;

Application Number: US19970852263 19970507 ;

Priority Number(s): US19970852263 19970507 ;

IPC Classification: G06F9/46 ;

Equivalents: GB2326255

ABSTRACT:

The automatic object distribution of the present invention allows object oriented programs to be run as distributed programs without any explicit networking code, and without using an interface definition language (IDL). The present invention allows programmers to experiment with different distributions without complicating the programming task. It accomplishes this by generating two proxies that allow method calls written for local invocation to be invoked over a network. These dynamically-generated proxies allow calls to flow across a network as if they were local.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平11－85519

(43)公開日　平成11年(1999)３月30日

(51)Int.Cl.[6]		識別記号	FI		
G06F	9/44	530	G06F	9/44	530M
	9/46	360		9/46	360B
	15/16	370		15/16	370N

審査請求 有　請求項の数9　OL　（全 9 頁）

(21)出願番号　特願平10－118308

(22)出願日　平成10年(1998)４月28日

(31)優先権主張番号　08／852263

(32)優先日　1997年５月７日

(33)優先権主張国　米国（US）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
INTERNATIONAL BUSINESS MASCHINES CORPORATION
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　デビッド・ルイス・カミンスキー
アメリカ合衆国27514、ノース・カロライナ州チャペル・ヒル、カーピン・ヒル・サークル　103

(74)代理人　弁理士　坂口　博　（外１名）

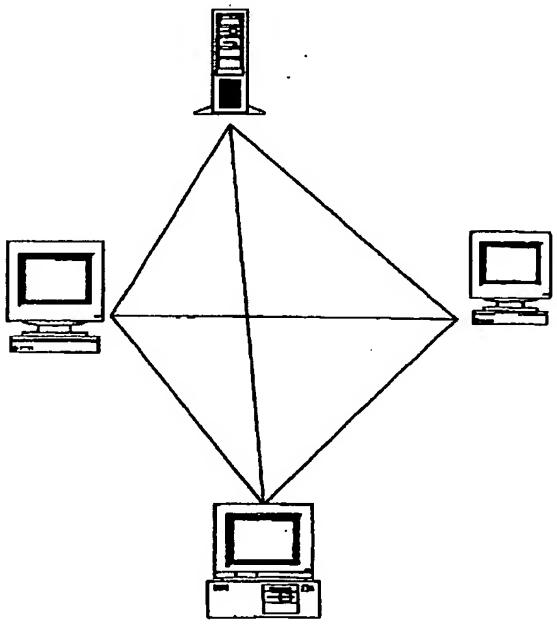
最終頁に続く

(54)【発明の名称】　オブジェクトを遠隔的に実行する方法、システム

(57)【要約】

【課題】　プログラマがプログラムを、あたかもそれが局所的に実行されているかのように作成することを可能にする、分散オブジェクト・プログラムを生成する改善された方法を提供すること。

【解決手段】　本発明の自動オブジェクト分散は、明示的なネットワーキング・コード無しに、またインタフェース定義言語（ＩＤＬ）を使用することなく、オブジェクト指向プログラムが分散プログラムとして実行されることを可能にする。本発明はプログラマがプログラミング・タスクを複雑化することなく、異なる分散を試すことを可能にする。これはローカル呼び出しのために作成されたメソッド呼び出しをネットワークを通じて呼び出すことを可能にする２つのプロキシを生成することにより達成される。これらの動的に生成されるプロキシが、呼び出しがあたかも局所的であるかのように、ネットワークを横断して伝番されることを可能にする。



【特許請求の範囲】
【請求項1】各々が1つ以上のプログラム・メソッドを含むプログラムのオブジェクトを、2つ以上の物理装置に渡り分散する方法であって、
前記プログラム内の全ての前記オブジェクトを識別するステップと、
第1のコンピュータ上に存在すべき前記オブジェクト、及び第2のコンピュータ上に存在すべき前記オブジェクトを決定するステップと、
リモート・コンピュータからアクセスされ得る各前記オブジェクトに含まれる全ての前記プログラム・メソッドを識別するステップと、
前記リモート・コンピュータからアクセスされ得る各前記オブジェクトに対して、第1のプロキシ及び第2のプロキシを生成するステップであって、前記第1のプロキシが前記第1のコンピュータ上に存在し、前記第2のプロキシが前記第2のコンピュータ上に存在し、前記第1のプロキシが、前記第2のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記第2のプロキシが、前記第1のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記プロキシを通じて、前記リモート・プログラム・メソッドをアクセスするステップと、
を含む、方法。
【請求項2】前記第2のプロキシが、前記第1のコンピュータ上に留まる前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含み、前記第1のプロキシが、前記第2のコンピュータに移動される前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含む、請求項1記載の方法。
【請求項3】前記ネットワーク・リンクが前記プログラム・メソッドのリモート呼び出しのためのステートメントを含む、請求項1記載の方法。
【請求項4】各々が1つ以上のプログラム・メソッドを含むプログラムのオブジェクトを、2つ以上の物理装置に渡り分散するコンピュータ・プログラム製品であって、コンピュータ読出し可能プログラム・コード手段が埋め込まれるコンピュータ読出し可能記憶媒体を含むものにおいて、前記コンピュータ読出し可能プログラム・コード手段が、
前記プログラム内の全ての前記オブジェクトを識別するコンピュータ読出し可能プログラム・コード手段と、
第1のコンピュータ上に存在すべき前記オブジェクト、及び第2のコンピュータ上に存在すべき前記オブジェクトを決定するコンピュータ読出し可能プログラム・コード手段と、
リモート・コンピュータからアクセスされ得る各前記オブジェクトに含まれる全ての前記プログラム・メソッドを識別するコンピュータ読出し可能プログラム・コード手段と、
前記リモート・コンピュータからアクセスされ得る各前記オブジェクトに対して、第1のプロキシ及び第2のプロキシを生成するコンピュータ読出し可能プログラム・コード手段であって、前記第1のプロキシが前記第1のコンピュータ上に存在し、前記第2のプロキシが前記第2のコンピュータ上に存在し、前記第1のプロキシが、前記第2のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記第2のプロキシが、前記第1のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、
前記プロキシを通じて、前記リモート・プログラム・メソッドをアクセスするコンピュータ読出し可能プログラム・コード手段と、
を含む、コンピュータ・プログラム製品。
【請求項5】前記第2のプロキシが、前記第1のコンピュータ上に留まる前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含み、前記第1のプロキシが、前記第2のコンピュータに移動される前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含む、請求項4記載のコンピュータ・プログラム製品。
【請求項6】前記ネットワーク・リンクが前記プログラム・メソッドのリモート呼び出しのためのステートメントを含む、請求項4記載のコンピュータ・プログラム製品。
【請求項7】各々が1つ以上のプログラム・メソッドを含むプログラムのオブジェクトを、2つ以上の物理装置に渡り分散するシステムであって、
前記プログラム内の全ての前記オブジェクトを識別する手段と、
第1のコンピュータ上に存在すべき前記オブジェクト、及び第2のコンピュータ上に存在すべき前記オブジェクトを決定する手段と、
リモート・コンピュータからアクセスされ得る各前記オブジェクトに含まれる全ての前記プログラム・メソッドを識別する手段と、
前記リモート・コンピュータからアクセスされ得る各前記オブジェクトに対して、第1のプロキシ及び第2のプロキシを生成する手段であって、前記第1のプロキシが前記第1のコンピュータ上に存在し、前記第2のプロキシが前記第2のコンピュータ上に存在し、前記第1のプロキシが、前記第2のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記第2のプロキシが、前記第1のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、
前記プロキシを通じて、前記リモート・プログラム・メソッドをアクセスする手段と、
を含む、システム。

【請求項8】前記第2のプロキシが、前記第1のコンピュータ上に留まる前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含み、前記第1のプロキシが、前記第2のコンピュータに移動される前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含む、請求項7記載のシステム。
【請求項9】前記ネットワーク・リンクが前記プログラム・メソッドのリモート呼び出しのためのステートメントを含む、請求項7記載のシステム。
【発明の詳細な説明】
【0001】
【発明の属する技術分野】本発明は一般に、コンピュータ・プログラミング方法及びシステムに関し、特にオブジェクト指向プログラミング、及びオブジェクト指向プログラムをネットワークにより接続される複数のコンピュータ上で実行する方法及びシステムに関する。
【0002】
【従来の技術】プログラムの一部が2つ以上のコンピュータ上で実行されるようにする、単一のプログラムの分散は、"デスクトップ"・コンピュータが強力になるにつれ、より一層普及するようになった。図1は、アプリケーションが実行される複数の分散コンピュータを含むコンピュータ・ネットワークの例を示す。大部分のコンピュータ・ネットワークは、より大規模な多くの階級が存在するが、この小さなネットワークが一例として使用される。現在、多くのコンピュータ・システムが、オブジェクトがネットワークを通じて通信することを可能にする。これを可能にする1つの方法は、RPC（Remote Procedure Call）すなわちリモート・プロシージャ呼び出しである。RPCは1980年代中旬から存在し、例えば、Birrell及びNelsonにより"Implementing Remote Procedure Calls"、ACM Transaction on Computer System 2、1984、pp39-59等で詳しく述べられている。更にDSOM（Distributed Systems Object Model）すなわち分散システム・オブジェクト・モデルなどの改善がRPCに対して行われ、現在に至っている。従来のシステムでは、プログラマがインタフェース定義言語（IDL：Interface Definition Language）を用いて、オブジェクトとのインタフェースを記述する。システムがIDL仕様を実行可能なコードに自動的に変換するツールを提供する。
【0003】最近、サン・マイクロシステムズ社が、RMI（Remote Method Invocation）すなわちリモート・メソッド呼び出しと呼ばれるRPC及びDSOMに類似する能力を含む、Javaと呼ばれるオブジェクト指向プログラミング言語を発売した。Javaが有効なシステムを用いることにより、プログラマはプログラムが実行されるネットワークの明示的な認識を有すること無く、分散オブジェクト・プログラムを作成することができる。しかしながら、この技術を用いてもプログラマまたはインプリメンタは、オブジェクトのリモート呼び出しを可能にする追加のコードを作成する（またはIDL仕様を作成する）ことを要求される。コーディング作業のために、IDLの作成は誤りを起こしがちである。更に、プログラマが、彼がオブジェクトを分散するために選択するポイントが次善であると判断する場合、彼は現仕様を消去し、別の仕様を作成しなければならない。最適な分散ポイントを選択することは困難な作業であり、この問題は最新のプログラミング環境においてしばしば発生する。更に、最適な分散ポイントは、コンピュータ・トポロジにもとづき変化し得る。例えば、2つのコンピュータが同等に強力な場合、最適な分散は各コンピュータ上で実行される等しい数のオブジェクトを有するであろう。また、2つのコンピュータが異なる能力を有する場合、より強力なコンピュータがより多くのオブジェクトを実行することになろう。異種のコンピュータ・トポロジが一般的であるので、IDLを用いて静的に作成されるコードは、しばしば次善の性能を生じる。
【0004】代わりに、多くのプログラム作成ツールにより、プログラマはツールを用いてプログラムを記入し、オブジェクトが分散されるオブジェクト境界を示すことができる。ツールは次にプログラムの実行可能コード生成する。しかしながら、これらのツールは静的プログラム・パーティションを生成し、従ってIDLシステムに類似の異種性の問題を被る。更に、ツールは通常、共通運用されない。従って、同一のプログラムに携わる複数のプログラマは一緒に効率的に作業するために、同一のツール上で標準化することを強要される。ツールは強みと弱みを有するので、単一のツール上での標準化は望ましくない。
【0005】
【発明が解決しようとする課題】本発明の目的は、プログラマがプログラムを、あたかもそれが局所的に実行されているかのように作成することを可能にする、分散オブジェクト・プログラムを生成する改善された方法を提供することである。
【0006】本発明の別の目的は、プログラマにより生成されたクラスを変更すること無しに、オブジェクトの分散を生成することである。
【0007】更に本発明の別の目的は、プログラム生成ツールの必要無しに、オブジェクトの分散を生成することである。
【0008】
【課題を解決するための手段】本発明は、オブジェクトを遠隔的に実行または走行するシステム、方法及びプログラム製品について述べ、これはここでは自動オブジェクト分散（AOD：Automatic Object Distribution）として参照される。図2に示されるように、オブジェクト指向プログラムは互いに呼び出しあうことにより、統一された機能を実行する多くの独立のオブジェクトによ

り作成される。図2は、コンピュータ・メモリ１０１を示し、そこにはオブジェクト１０２、すなわちV、W、X、Y及びZの全てが内在する。本発明は、ユーザがオブジェクトの一部を第1のコンピュータから第2のコンピュータに移動して、実行することを可能にする。この様子が図3に示される。図3では、オブジェクトV及びYすなわち２０５が、第1のコンピュータ・メモリ２０１内に留まり、オブジェクトW、X及びZすなわち２０７が、第2のコンピュータ・メモリ２０３に移動される。これは例えば、第1のコンピュータと第2のコンピュータとの間の作業負荷を平衡化するために行われる。このことは追加の複雑化を生じる。なぜなら、プログラム・オブジェクトの各々が、ここではメソッドとして参照される外部ルーチンを呼び出すことができるからである。プログラム・オブジェクトをその実行のために複数のコンピュータに分離することは、全てのメソッドまたはオブジェクトが（これらはオブジェクトの異なるコンピュータ・メモリへの分離により、もはや同一システム上に呼び出し側メソッドと共存しないメソッドをアクセスする）、オブジェクトの一部がネットワーク上のどこか別のところに配置されていることを知り、それらのリモート・オブジェクトをアクセスする方法を理解することを要求する。

【０００９】本発明のＡＯＤは、プログラマがオブジェクトがネットワーク上で分散されると言う知識を有して、彼らのプログラムを作成する要求を排除する。本ＡＯＤはまた、プログラマが分散オブジェクトをサポートする特殊な言語により、プログラムを作成する、或いは作業をネットワークにまたがり分散したいときに、特定のツールを用いて作成する必要性を排除する。代わりに、本発明を用いることにより、プログラマはＪａｖａ（サン・マイクロシステムズ社の商標）などの共通のオブジェクト指向言語によりプログラムを作成し、コードをリンクされていない実行可能コードにコンパイルする。（コンピュータ文献では、リンクされていない実行可能コードは、しばしば"オブジェクト・コード"と呼ばれる。オブジェクト指向システムを述べるとき、用語"オブジェクト・コード"の使用は混乱を生じ、従って用語"バイトコード"が本発明の目的のために採用される。）プログラムはコードをあたかもプログラムが単一のマシン上で実行されるように、正確に作成する。すなわち、彼はプログラムをＩＤＬシステムの場合のように、追加の情報により修飾しない。プログラムは次に、分散が発生すべき場所を決定し、ＡＯＤが分散コードを生成する。

【００１０】ＡＯＤは2つのプロキシを生成することにより分散を実行する。プロキシは、ローカル呼び出しのために作成されるメソッド呼び出しが、ネットワークを通じて呼び出されることを可能にする。局所的に実行されるあるクラスYと、遠隔的に実行されるあるクラスXとの間で分割が示される場合、ＡＯＤプロセスはプロキシを生成して介在するネットワークを克服する。これらのプロキシは一緒にYからの呼び出しを横取りし、それらをXに受け渡し、結果をYに返却する。この様子が図４及び図５に詳細に示される。

【００１１】図4は、プログラムが単一メモリ３０１内で実行されるコンピュータ・システムを示す。オブジェクトＹ３０３及びオブジェクトＸ３０５の両方は、同一のメモリ上に存在し、オブジェクトＹ３０３からオブジェクトＸ３０５を呼び出すために、メソッド呼び出し３１１が使用される。

【００１２】図5に示されるように、ＡＯＤプロセスはバイトコード・ファイル内の情報を用い、あるクラス例えばＸ３０５のために、そのプロキシとして機能するクラスを含む2つのプロキシ・ファイル（X'３１５及びX"３１７）を生成する。X'３１５は、Ｘ３０５内の全ての公開メソッドを有するXと命名されるクラスを含む。（オリジナル・クラスＸ３０５ファイルはリモート・マシン上に配置されるので、名前の衝突は存在しない。）X"３１７は、固有の名前（X"）を有するクラスを含み、これはクラスX内の公開メソッドへの呼び出しを生成するメソッドを含む。X'３１５はＹ３０３と同一のマシン上に存在し、X"３１７はＸ３０５と同一のマシン上に存在する。X'３１５はXと呼ばれるクラスを含み、そのクラスはＸ３０５内の全ての公開メソッドを含むので、Ｙ３０３がオリジナル・クラスX内のメソッドに対する呼び出しを生成するとき、その呼び出しは実際には、ファイルX'３１５内に含まれ、局所的に存在する新たなクラスX内のメソッドに対して生成される。X'３１５は呼び出し情報をX"３１７に受け渡し、X"３１７が次にＸ３０５にローカル呼び出しを生成する。ローカル呼び出しの結果がX"３１７及びX'３１５を介してＹ３０３に伝播される。結果的に、プログラマが任意のプログラム・コードを作成または変更することを要求すること無く、局所的に実行されるように作成されたアプリケーションが、ネットワークを通じて分散される。

【００１３】

【発明の実施の形態】本発明の好適な実施例について、図面を参照しながら説明する。しかしながら、本発明はこの好適な実施例に限られるものではない。本発明は、プロキシを使用する自動分散プログラミングに適用可能である。好適な実施例の自動オブジェクト分散（ＡＯＤ）は、Ｊａｖａプログラミング環境を使用するが、本発明はＪａｖａに限られるものではなく、本発明の他のオブジェクト指向環境への適用が当業者には理解されよう。本発明の好適な実施例は、あたかもプログラムが局所的に実行されるかのように（例えばプログラムがリモート・メソッド呼び出しコードを含まない）、プログラマがプログラムをＪａｖａにより作成し、標準のＪａｖ

aコンパイラを用いてソース・ファイルを対応するバイトコード・ファイルにコンパイルしたものと仮定する。AODプロセスは次に、オブジェクト指向プログラミングにおいては、全てのオブジェクトが（公開変数への直接アクセスを通じてではなく）、メソッド呼び出しを通じてアクセスされると言う取り決めを利用する。Javaでは、各クラスが別々のバイトコード・ファイルにコンパイルされる（従って、バイトコード・ファイルはしばしば"クラス・ファイル"と呼ばれる）。プログラマは次に、リモート・マシンに移動されるクラス及び局所的に留まるクラスを識別する。こうしたクラスを識別するプロセスは、クラスを含むファイルの名前を本発明を実現するツールにタイプ入力することにより達成される。

【0014】本発明の好適な実施例を用いて、オブジェクトを遠隔的に実行するためにAODはどのクラスが局所的に維持され、どのクラスが遠隔的に移動されるかに関する入力を用いて、図6に言及される次の処理を実行する。

1）AODは各クラスに対するバイトコードを検査し、遠隔的に実行されるクラスからメソッド呼び出しを受信するクラスを決定する。（ステップ502）

2）リモート・マシンから要求される各オブジェクトXに対して、その公開メソッドのリストがXのバイトコード・ファイルから抽出され、記憶される。（ステップ504）

3）公開メソッドのリストにもとづき、X"と呼ばれるプロキシが生成される。X"は、X内の公開メソッドと類似の名前を有するメソッドを含むように構成される。構成されるメソッドの各々はXに対する対応するメソッド呼び出しを生成する。それはまた、自身をRMIレジストリに登録するために必要なコードを含む。（ステップ506）

4）公開メソッドのリストにもとづき、X'と呼ばれるプロキシが生成される。X'はX内の公開メソッドを含むように構成される。構成されるメソッドの各々は、（JavaのRMI機能を用い）X"に対する対応するリモート・メソッド呼び出しを生成する。このプロキシ・クラスは、Xが有するのと同一の名前を有する。（ステップ508）

5）リモートとして指定される全てのオリジナル・バイトコード・ファイル、及び全ての"2重プライム符号付きプロキシ"がリモート・マシンに転送される。一方、ローカルとして指定される全てのバイトコード・ファイル及び全ての"プライム符号付き"プロキシは、ローカル・マシン上に留まる。（ステップ510）

【0015】各ステップについて、より詳細に述べることにする。

1）Javaコンパイラ（及び多くのオブジェクト指向言語のコンパイラ）は、検査プロセスが成功裡に発生することを可能にする、公知の形式のバイトコード・ファイルを生成する。様々なバイトコード・ファイルのリンクを可能にするため、各バイトコード・ファイルは各クラスから呼び出されるメソッドに関する情報を含む。従って、ファイルからメソッドに関する情報を読出すことにより、AODは相関するクラスの要求リストを生成することができる。

【0016】2）同様に、バイトコード・ファイルは、それにより記述されるクラスの公開メソッドに関する情報を含む。従って、ファイルから公開メソッドに関する情報を読出すことにより、AODは全ての公開メソッドのリストを生成することができる。バイトコード・ファイルはまた、パラメータ及び戻り値に関する情報を含む。

【0017】Javaの一部の実施例では、クラスによりサポートされる公開メソッドのプログラム的決定を可能にする"リフレクション（reflection）"と呼ばれる機能を含む。リフレクションが実施される場合、これはバイトコード・ファイルを検査するプロセスを置換するために使用される。

【0018】3）クラス上の全ての公開メソッドのリストが提供されると、類似に命名されるメソッドを含む2重プライム符号付きプロキシ・クラス（X"）を生成することは容易であり、これらのメソッドの各々がX上のメソッドへの呼び出しを生成する。好適な実施例では、呼び出しはJavaソースとして生成され、Javaの標準の動的コンパイル機構を用いて、動的にコンパイルされる。従って、XがメソッドA及びBを含む場合、X"は次のように作成される。

```
Class Xprimeprime extends UnicastRemoteObject
                    implements XprimeprimeInterface{
Xx;
Xprimeprime(){//initializer
 x=new X();
}
A_AOD(){//passes through calls to A
 x.A();
}
B_AOD(){//passes through calls to B
 x.B();
}
```

【0019】ここで、プロキシ内のメソッドに"_AOD"の接尾部を付加する取決めを用いた点に注意されたい。例えばメソッド"A_AOD"は、X内のメソッド"A"を呼び出す。再生可能で一貫性があれば、任意の取り決めが使用され得る。

【0020】4）クラス上の全ての公開メソッドのリストが提供されると、これらのメソッドを含むプライム符号付きプロキシ・クラスを生成することは容易であり、これは（RMIを用いて、）X"上で類似に命名されたメソッドへのリモート呼び出しを生成する。好適な実施

例では、呼び出しがＪａｖａ　ＲＭＩソースとして生成され、Ｊａｖａの標準の動的コンパイル機構を用いて動的にコンパイルされる。

【００２１】プロキシX"に加え、X内の全ての公開メソッドの署名を含むインタフェースX"インタフェースが、Ｊａｖａの標準ＲＭＩ技法により要求されるときに生成される。

```
Interface XprimeprimeInterface extends java.rmi.Remote{
  A_AOD();
  B_BOD();
}
```

【００２２】クラス上の全ての公開メソッドに関する同一の情報が提供されると、インタフェースX"インタフェースを生成することも容易である。

【００２３】

【００２４】従って、XがA及びBを含む場合、X'は次のように生成される。

```
Class Xprime{
  XprimeprimeInterface x;
  Xprime(){//constructor
  x=(XprimeprimeInterface)Naming.lookup(Xprimeprime);//do the RMI lookup
  }
  A(){//call remote A
   x.A_AOD();
  }
  B(){//call remote B
   x.B_AOD();
  }
}
```

【００２５】この例では、Naming.lookup()がＲＭＩレジストリに問い合わせることにより、リモート参照をX"に返却する。取り決めにより、返却オブジェクトがX"インタフェースのタイプに投じられる。

【００２６】プロキシ・クラスはXと呼ばれるが、オリジナルXとの名前の衝突は発生しない。プロキシXはローカル・マシン上で実行され、オリジナルXはリモート・マシンに移動すると識別されたので、１マシン当たり１つのXだけが存在する。

【００２７】５）ファイルが次に、この機能が実現されるシステム上で使用可能な任意の手段により転送される。

【００２８】図5に示されるように、プロキシが確立されると、次のステップが実行時に発生する。：プログラマがあるオブジェクトYとあるオブジェクトXとの間で、プログラムを分散するように選択すると仮定しよう。ここでYはメソッド呼び出しを通じてのみ、Xをアクセスする。YがX上の公開メソッドfoo()を呼び出すと、次のステップが発生する。最初に、Yが通常通りfoo()への呼び出しを生成する。（そのコードが変更されていないので必然である。）プロキシXprimeはローカル・マシン上の唯一のXであり、メソッドfoo()を含むので、呼び出しはXのプライム符号付きプロキシ上で生成されなければならない。

【００２９】取り決めによりX内のメソッドfoo()は、リモート２重プライム符号付きプロキシ上のメソッドfoo_AOD()へのＲＭＩ呼び出しを生成し、結果を返却する。取り決めにより、２重プライム符号付きプロキシ上のfoo_AOD()は、X上のfoo()へのローカル呼び出しを生成する。Xは変更されていないので、これはオリジナルfoo()を実行し、結果を２重プライム符号付きプロキシ内のfoo_AOD()に送り返す。すると、foo_AOD()はそれらをプライム符号付きプロキシに渡し、プライム符号付きプロキシがそれらをYに返却する。従って、呼び出しは意味的にはローカル呼び出しに等価である。

【００３０】このプロセスはオリジナル・ファイルに影響を及ぼさず、いずれのプログラマ介入も要求することなく分散を実現するので、プログラマにとって異なるクラス分散ポイントをテストすることが極めて容易となる。このことがプログラマに多大な影響を及ぼすこと無く、プログラムをより効率的に実行することを可能にする。

【００３１】更に、分散は追加のプログラミングを要求しないので、システム管理者は同一のプログラムの複数のインスタンスを、異なるコンピュータ構成上に配置することができ、各構成はコンピュータ環境のセグメントの特定の特性にもとづき、それ自身の分散（及び性能特性）を有することになる。この分散は、追加のプログラミングを必要とせず、またオリジナル・プログラムに対するいずれの変更も要求しない。

【００３２】まとめとして、本発明の構成に関して以下の事項を開示する。

【００３３】（１）各々が１つ以上のプログラム・メソッドを含むプログラムのオブジェクトを、２つ以上の物

理装置に渡り分散する方法であって、前記プログラム内の全ての前記オブジェクトを識別するステップと、第1のコンピュータ上に存在すべき前記オブジェクト、及び第2のコンピュータ上に存在すべき前記オブジェクトを決定するステップと、リモート・コンピュータからアクセスされ得る各前記オブジェクトに含まれる全ての前記プログラム・メソッドを識別するステップと、前記リモート・コンピュータからアクセスされ得る各前記オブジェクトに対して、第1のプロキシ及び第2のプロキシを生成するステップであって、前記第1のプロキシが前記第1のコンピュータ上に存在し、前記第2のプロキシが前記第2のコンピュータ上に存在し、前記第1のプロキシが、前記第2のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記第2のプロキシが、前記第1のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記プロキシを通じて、前記リモート・プログラム・メソッドをアクセスするステップと、を含む、方法。

（2）前記第2のプロキシが、前記第1のコンピュータ上に留まる前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含み、前記第1のプロキシが、前記第2のコンピュータに移動される前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含む、前記（1）記載の方法。

（3）前記ネットワーク・リンクが前記プログラム・メソッドのリモート呼び出しのためのステートメントを含む、前記（1）記載の方法。

（4）各々が1つ以上のプログラム・メソッドを含むプログラムのオブジェクトを、2つ以上の物理装置に渡り分散するコンピュータ・プログラム製品であって、コンピュータ読出し可能プログラム・コード手段が埋め込まれるコンピュータ読出し可能記憶媒体を含むものにおいて、前記コンピュータ読出し可能プログラム・コード手段が、前記プログラム内の全ての前記オブジェクトを識別するコンピュータ読出し可能プログラム・コード手段と、第1のコンピュータ上に存在すべき前記オブジェクト、及び第2のコンピュータ上に存在すべき前記オブジェクトを決定するコンピュータ読出し可能プログラム・コード手段と、リモート・コンピュータからアクセスされ得る各前記オブジェクトに含まれる全ての前記プログラム・メソッドを識別するコンピュータ読出し可能プログラム・コード手段と、前記リモート・コンピュータからアクセスされ得る各前記オブジェクトに対して、第1のプロキシ及び第2のプロキシを生成するコンピュータ読出し可能プログラム・コード手段であって、前記第1のプロキシが前記第1のコンピュータ上に存在し、前記第2のプロキシが前記第2のコンピュータ上に存在し、前記第1のプロキシが、前記第2のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記第2のプロキシが、前記第1のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記プロキシを通じて、前記リモート・プログラム・メソッドをアクセスするコンピュータ読出し可能プログラム・コード手段と、を含む、コンピュータ・プログラム製品。

（5）前記第2のプロキシが、前記第1のコンピュータ上に留まる前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含み、前記第1のプロキシが、前記第2のコンピュータに移動される前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含む、前記（4）記載のコンピュータ・プログラム製品。

（6）前記ネットワーク・リンクが前記プログラム・メソッドのリモート呼び出しのためのステートメントを含む、前記（4）記載のコンピュータ・プログラム製品。

（7）各々が1つ以上のプログラム・メソッドを含むプログラムのオブジェクトを、2つ以上の物理装置に渡り分散するシステムであって、前記プログラム内の全ての前記オブジェクトを識別する手段と、第1のコンピュータ上に存在すべき前記オブジェクト、及び第2のコンピュータ上に存在すべき前記オブジェクトを決定する手段と、リモート・コンピュータからアクセスされ得る各前記オブジェクトに含まれる全ての前記プログラム・メソッドを識別する手段と、前記リモート・コンピュータからアクセスされ得る各前記オブジェクトに対して、第1のプロキシ及び第2のプロキシを生成する手段であって、前記第1のプロキシが前記第1のコンピュータ上に存在し、前記第2のプロキシが前記第2のコンピュータ上に存在し、前記第1のプロキシが、前記第2のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記第2のプロキシが、前記第1のコンピュータ上の前記プログラム・メソッドをアクセスするためのネットワーク・リンク及び指示を含み、前記プロキシを通じて、前記リモート・プログラム・メソッドをアクセスする手段と、を含む、システム。

（8）前記第2のプロキシが、前記第1のコンピュータ上に留まる前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含み、前記第1のプロキシが、前記第2のコンピュータに移動される前記オブジェクト内の前記メソッドに類似に命名されるメソッドを含む、前記（7）記載のシステム。

（9）前記ネットワーク・リンクが前記プログラム・メソッドのリモート呼び出しのためのステートメントを含む、前記（7）記載のシステム。

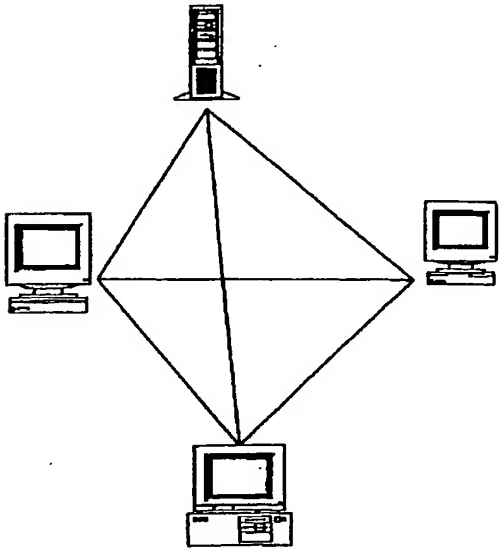
【図面の簡単な説明】

【図1】本発明が実施され得るネットワーク例を示す図である。

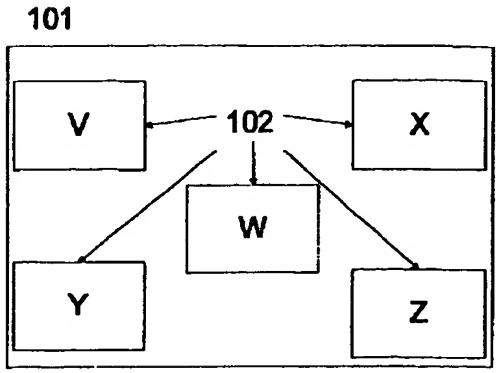
【図2】5つのオブジェクトを含むコンピュータ・メモ

リを示す図である。
【図3】2つのオブジェクトがあるコンピュータ・メモリ内で実行され、3つのオブジェクトが異なるコンピュータ・メモリ内で実行される分散システムを示す図である。
【図4】2つのオブジェクトが同一のメモリ内実行される方法を示す図である。
【図5】本発明に従い分散された後の、図4のオブジェクトを示す図である。
【図6】本発明の分散方法のフローチャートを示す図である。
【符号の説明】
101、201、203 、301、302 コンピュータ・メモリ
102、205、207、303、305 オブジェクト
311 メソッド呼び出し
315、317 プロキシ・ファイル

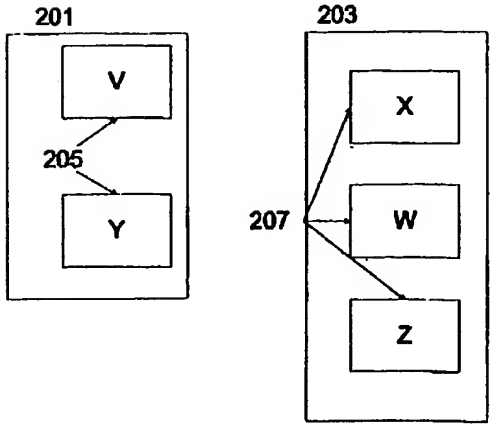
【図1】



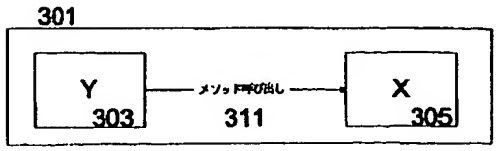
【図2】



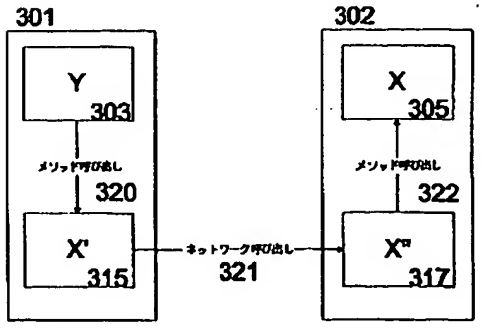
【図3】



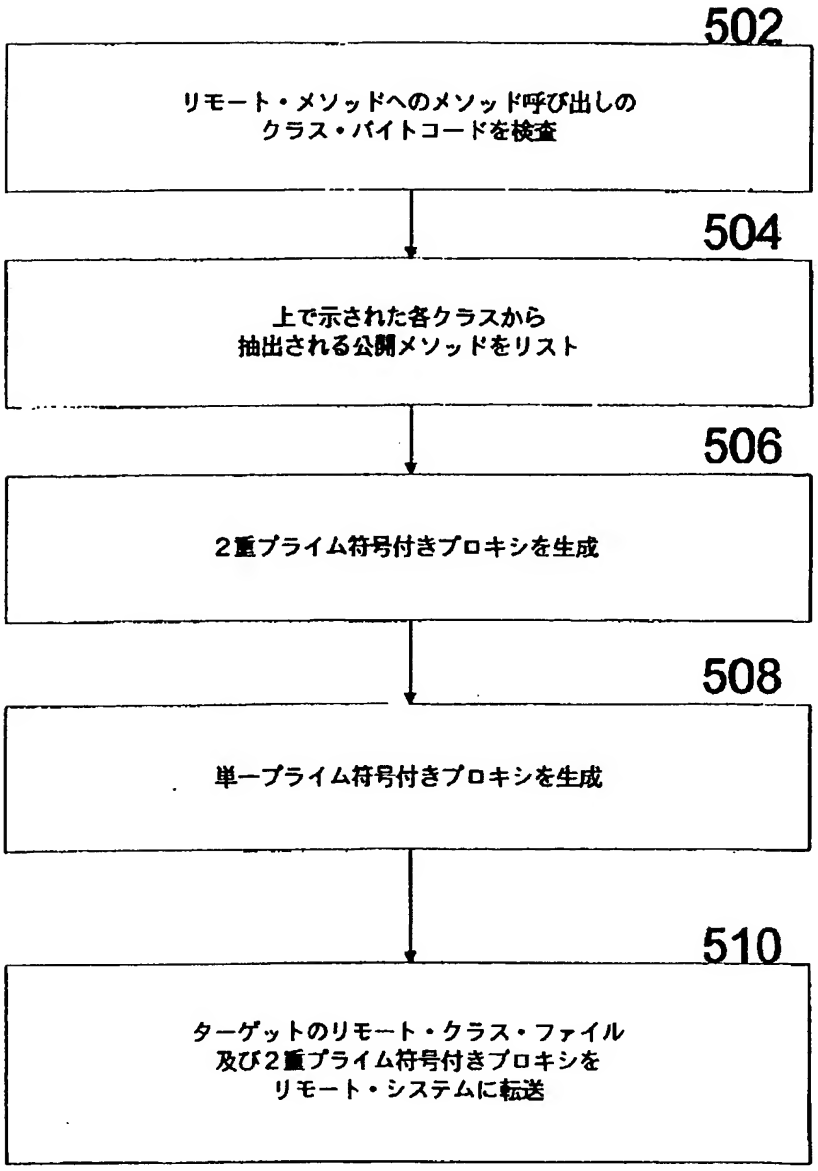
【図4】



【図5】



【図6】



フロントページの続き

(72)発明者　リチャード・アダム・キング
アメリカ合衆国27514、ノース・カロライナ州チャペル・ヒル、エクスカリバー・コート　214

(72)発明者　ツィーヨン・リ
アメリカ合衆国27705、ノース・カロライナ州ダラム、ルイス・サークル　863